Trabajo Práctico Especial 1: Reservas de Atracciones de Parques Temáticos

30 de Agosto de 2023

Objetivo

Implementar en grupos un **sistema remoto *thread-safe*** para la **reserva de atracciones de un parque temático en un año**, permitiendo notificar a los usuarios del servicio y ofreciendo reportes de las reservas realizadas al momento.

Descripción Funcional

Servicios Remotos

El sistema requiere el desarrollo de **los siguientes servicios remotos**:

1. Servicio de Administración del Parque

- Funcionalidad: Administrar las atracciones y pases de atracciones de un parque para un año determinado
- Usuario: Empresa administradora del parque
- Debe contar con métodos para:
    - **1.1 Agregar una atracción** al parque a partir de su **nombre**, sus **horarios de apertura y cierre** y la cantidad de **minutos que tendrán los "*slot*"** para poder realizar reservas. Falla si existe una atracción con ese nombre, si los valores de los horarios son inválidos, si los minutos no son positivos o si con los valores provistos no existe un slot posible.
      Cuando los visitantes del parque reserven una atracción deberán indicar un ***slot***.
      Por ejemplo, si los *slot* de una atracción son de 15 minutos y el horario de apertura de la atracción es 08:00 y el de cierre es 20:00 entonces los *slot* para reservas son 08:00, 08:15, 08:30, … 19:30 y 19:45. Según la combinación de valores de horarios y minutos de slot puede darse el escenario de que el último slot no tenga la cantidad de minutos esperada y es válido (por ejemplo si abre a las 09:00 y cierra a las 10:00 y los slots son de 40 minutos).
    - **1.2 Agregar un pase de atracciones** vendido a partir del id del **visitante** (en formato UUID), el **tipo** de pase y el **día** del año de validez del pase.
      Cuando los visitantes del parque reserven una atracción deberán contar con un pase válido para el día del año indicado. Existen tres tipos de pase:
        i. **UNLIMITED**: Para reservar una cantidad ilimitada de atracciones del parque
        ii. **THREE**: Para reservar hasta 3 atracciones del parque (hasta tres reservas pendientes o confirmadas)
        iii. **HALF_DAY:** Para reservar una cantidad ilimitada de atracciones del parque para los *slots* que inicien antes de las 14:00.

      Falla si ya existe un pase para el visitante para el día indicado o si el tipo de pase es inválido o si el día del año es inválido.

- **1.3 Cargar la capacidad de los *slots* de una atracción** a partir del **nombre** de la atracción, un **día** del año y una **capacidad**. Falla si la atracción no existe, si el día es inválido, si la capacidad es negativa o si ya se cargó una capacidad para esa atracción y día.
  La capacidad aplica a todos los *slots* del día indicado para la atracción indicada. Esta carga confirmará, reubicará o cancelará las reservas con estado pendiente que ya se hayan realizado para esta atracción para el día indicado. Para cada slot del día (en orden cronológico) y en función de la capacidad indicada se deberán primero confirmar todas las reservas posibles teniendo en cuenta el orden de la realización de la reserva (priorizando a las primeras N reservas realizadas siendo N la capacidad del slot). Luego, para cada slot del día (en orden cronológico), y de existir todavía reservas pendientes, se deberán intentar reubicar esas reservas en otros slots "cercanos" del mismo día, siempre considerando la capacidad de cada slot. Se deberán intentar todos los slots posteriores al de la reserva original. Por ejemplo, y suponiendo slots de 15 minutos, si se contaba con una reserva pendiente para las 15:30 que no se pudo confirmar en ese horario entonces se deberá primero intentar reubicar la reserva en el slot 15:45. Si este slot ya cuenta con una cantidad de reservas (confirmadas y/o pendientes) mayor o igual a la capacidad del slot entonces se procede a intentar con el slot siguiente (el de las 16:00) y así hasta el último slot del día. Estas reservas reubicadas mantienen el estado de pendiente y deberán ser confirmadas manualmente por el visitante más tarde utilizando el método correspondiente del servicio. Por último se deben cancelar las reservas pendientes que no pudieron ser reubicadas. Al final del proceso, se debe indicar cuántas reservas pendientes fueron confirmadas, reubicadas o canceladas.

2. Servicio de Reserva de Atracciones

- Funcionalidad: Reservar atracciones del parque
- Usuario: Visitante / Usuario Final de la App Mobile de Reserva de Atracciones
- Debe contar con métodos para:
  - **2.1 Consultar las atracciones** del parque, indicando para cada atracción el nombre, el horario de apertura y el horario de cierre.
  - **2.2 Consultar la disponibilidad de las atracciones**, indicando la cantidad de reservas (pendientes y confirmadas) y la capacidad de los slots de la atracción (si se cargó), a partir de uno de los siguientes criterios:
    i. Un slot de una atracción a partir del **día** del año, el nombre de la **atracción** y el **slot** (en formato HH:MM)
    ii. Un rango de slots de una atracción a partir del **día** del año, el nombre de la **atracción** y los **dos valores de slot** que definen el rango (ambos en formato HH:MM). Por ejemplo, si los slot de una atracción son de 15 minutos y se indica el rango [15:30, 16:00] se debe listar la disponibilidad de los slots 15:30, 15:45 y 16:00.
    iii. Un rango de slots de todas las atracciones del parque a partir del **día** del año y los **dos valores de slot** que definen el rango (ambos en formato HH:MM).

    Falla (según los criterios indicados) si no existe una atracción con ese nombre, si el día es inválido o si el slot o rango de slot es inválido.
  - **2.3 Reservar una atracción** a partir del nombre de la **atracción**, el **día** del año, el **slot** de la reserva (en formato HH:MM) y el **id** del visitante (en formato UUID). Falla

si la reserva ya existe, si no se puede reservar la atracción según las restricciones del tipo de pase, si no existe una atracción con ese nombre, si el día es inválido, si el slot es inválido o si no cuenta con un pase válido para ese día. Retorna el tipo de reserva realizada:

i. **Si no se cargó aún la capacidad de los slot** de la atracción entonces se realiza la reserva con **estado pendiente**. Se pueden acumular sobre la atracción reservas pendientes sin límite.
ii. **Si ya se cargó la capacidad de los slot** de la atracción entonces se realiza la reserva con **estado confirmada**. Se pueden acumular sobre la atracción una cantidad de reservas confirmadas menor o igual a la capacidad del slot. Falla si se intenta reservar y ya se alcanzó la capacidad.

Es posible realizar múltiples reservas para una misma atracción en el mismo día y múltiples reservas que tengan el mismo "*slot*".

- **2.4 Confirmar una reserva pendiente de una atracción** a partir del nombre de la **atracción**, el **día** del año, el **slot** de la reserva (en formato HH:MM) y el **id** del visitante (en formato UUID). Falla si no se cargó la capacidad de los slots de la atracción para ese día, si la reserva ya está confirmada, si no existe una reserva realizada para la atracción con ese pase, si no existe una atracción con ese nombre, si el día es inválido, si el slot es inválido o si no cuenta con un pase válido para ese día.
- **2.5 Cancelar una reserva de una atracción** a partir del nombre de la **atracción**, el **día** del año, el **slot** de la reserva (en formato HH:MM) y el **id** del visitante (en formato UUID). Se pueden cancelar tanto reservas pendientes como confirmadas. Falla si no existe una reserva realizada para la atracción con ese pase, si no existe una atracción con ese nombre, si el día es inválido, si el slot es inválido o si no cuenta con un pase válido para ese día.

3. Servicio de Notificaciones de una atracción

- Funcionalidad: Recibir notificaciones respecto a los cambios en las reservas de una atracción del parque
- Usuario: Visitante / Usuario Final de la App Mobile de Reserva de Atracciones
- Debe contar con métodos para:
  - **3.1 Registrar a un visitante para que sea notificado de los eventos relacionados a una reserva**, a partir del nombre de la **atracción**, el id del **visitante** (en formato UUID) y el **día** del año del pase. Falla si no existe una atracción con ese nombre, si el día es inválido, si no cuenta con un pase válido para ese día o si ya estaba registrado para ser notificado de esa atracción ese día. Se considera un evento:
    i. Se registró correctamente, informando el estado de la reserva en ese momento
    ii. Se cargó la capacidad de los slot de la atracción para ese día
    iii. Se confirmó una reserva pendiente del visitante en la atracción para ese día
    iv. Se reubicó una reserva pendiente del visitante en la atracción para ese día
    v. Se canceló una reserva pendiente del visitante en la atracción para ese día
  - **3.2 Anular el registro de un visitante para no recibir más notificaciones** de la atracción, a partir del nombre de la **atracción**, el id del **visitante** (en formato UUID) y el **día** del año del pase. Falla si no está registrado para recibir notificaciones de una atracción, si no existe una atracción con ese nombre, si el día es inválido o si no cuenta con un pase válido para ese día.

4. Servicio de Consulta

- Funcionalidad: Consultar las reservas pendientes y confirmadas de las atracciones
- Usuario: Empresa administradora del parque
- Debe contar con métodos para:
    - **4.1 Consultar la capacidad sugerida, en base a las reservas pendientes, para todas las atracciones del parque para un día**, indicando para cada atracción:
        i. Nombre de la atracción
        ii. Capacidad sugerida, que consiste en la cantidad máxima de reservas pendientes para todos los slots del día de la atracción.
        iii. Slot correspondiente a esa cantidad máxima (en formato HH:MM)

      en orden descendente por la capacidad sugerida, a partir del **día** del año. Falla si el día es inválido. Si la atracción ya cuenta con una capacidad cargada entonces no debe listarse en la consulta.
    - **4.2 Consultar las reservas confirmadas para todas las atracciones del parque para un día**, indicando para cada reserva:
        i. Nombre de la atracción
        ii. Id del visitante (en formato UUID)
        iii. Slot (en formato HH:MM)

      en orden de confirmación de la reserva, a partir del **día** del año. Falla si el día es inválido.

Clientes

Para poder probar el sistema se requiere que se implementen cuatro programas cliente, cada uno coincidente con cada servicio remoto.

Cliente de Administración del Parque

La información de cuál es la acción a realizar se recibe a través de argumentos de línea de comando al llamar al *script* del cliente de administración del parque y el resultado se debe imprimir en pantalla.

Por ejemplo:

```
$> ./admin-cli -DserverAddress=xx.xx.xx.xx:yyyy -Daction=actionName
[ -DinPath=filename | -Dride=rideName | -Dday=dayOfYear | -Dcapacity=amount ]
```

donde

- **`admin-cli`** es el *script* que invoca a la clase del cliente de administración del parque
- `xx.xx.xx.xx:yyyy` es la dirección IP y el puerto donde está publicado el servicio de administración del parque
- `actionName` es el nombre de la acción a realizar
    - **`rides`**: Agrega un lote de atracciones (ver detalle más abajo)
    - **`tickets:`** Agrega un lote de pases (ver detalle más abajo)
    - **`slots:`** Carga la capacidad `amount` de los *slots* de la atracción con nombre `rideName` para el día del año `dayOfYear`.

Para todas las acciones se deberá imprimir en pantalla el resultado de la operación o el error correspondiente.

De esta forma,

```
$> ./admin-cli -DserverAddress=10.6.0.1:50051 -Daction=slots -Dday=100
-Dride=SpaceMountain -Dcapacity=30
```

carga una capacidad de `30` visitantes por slot a la atracción `SpaceMountain` para el día `100` del año utilizando el servicio remoto publicado en `10.6.0.1:50051` e imprime en pantalla el resultado de la operación y un detalle del resultado de la operación:

```
Loaded capacity of 30 for SpaceMountain on day 100
12 bookings confirmed without changes
4 bookings relocated
2 bookings cancelled
```

Acción `rides`

El cliente deberá leer de un archivo CSV un **lote de atracciones** con los siguientes campos (delimitados por un ";"):

- **Nombre** (String)
- **Horario de Apertura** (HH:MM)
- **Horario de Cierre** (HH:MM)
- **Cantidad de minutos por *slot*** (Entero positivo)

Cada línea del archivo representa una atracción. El archivo contendrá una primera línea de encabezado.

Ejemplo de las primeras tres líneas de archivo:

```
name;hoursFrom;hoursTo;slotGap
SpaceMountain;09:00;22:00;30
TronLightcycle;10:00;22:00;15
```

- En la segunda línea se indica que se quiere agregar la atracción `SpaceMountain` que abre a las 09:00 y cierra a las 22:00, donde cada slot tiene 30 minutos. Los slot serán entonces 09:00, 09:30, 10:00, …, 21:00 y 21:30.

El *path* del archivo necesario para cargar un lote de atracciones se recibe con `-DinPath=filename` y el resultado se debe imprimir en pantalla.
De esta forma,

```
$> ./admin-cli -DserverAddress=10.6.0.1:50051 -Daction=rides
-DinPath=../attractions.csv
```

realiza la carga de las atracciones presentes en el archivo que se encuentra en el *path* `../attractions.csv`, utilizando el servicio remoto publicado en `10.6.0.1:50051` e imprime en pantalla un mensaje luego de haber realizado todas las solicitudes. Por ejemplo, para un archivo `attractions.csv` de 11 renglones se imprime:

```
10 attractions added
```

En el caso de que la carga del lote de atracciones ocasione un error, se debe imprimir el detalle de las atracciones que no pudieron agregarse. Por ejemplo, para un archivo que contenga cinco atracciones donde dos de ellas cuentan con el mismo nombre se imprime:

```
Cannot add 1 attractions
4 attractions added
```

Acción `tickets`

El cliente deberá leer de un archivo CSV un **lote de pases** con los siguientes campos (delimitados por un ";"):

- **Id del visitante** (UUID)
- **Tipo de Pase** (String)
- **Día del año** de validez (Entero positivo)

Cada línea del archivo representa un pase. El archivo contendrá una primera línea de encabezado.

Ejemplo de las primeras tres líneas de archivo:

```
visitorId;passType;dayOfYear
ca286ef0-162a-42fd-b9ea-60166ff0a593;UNLIMITED;100
2af16ea7-4af1-47f6-bf46-8515de5a500f;HALFDAY;15
```

- En la segunda línea se indica que se vendió un pase ilimitado (`UNLIMITED`) para el visitante de id `ca286ef0-162a-42fd-b9ea-60166ff0a593` válido para el día 100 del año.

El *path* del archivo necesario para cargar un lote de pases se recibe con `-DinPath=filename` y el resultado se debe imprimir en pantalla.
De esta forma,

```
$> ./admin-cli -DserverAddress=10.6.0.1:50051 -Daction=tickets
-DinPath=passes.csv
```

realiza la carga de los pases presentes en el archivo `passes.csv`, utilizando el servicio remoto publicado en `10.6.0.1:50051` e imprime en pantalla un mensaje luego de haber realizado todas las solicitudes. Por ejemplo, para un archivo `passes.csv` de 101 renglones se imprime:

```
100 passes added
```

En el caso de que la carga del lote de pases ocasione un error, se deben imprimir el detalle de los pases que no pudieron agregarse. Por ejemplo, para un archivo que contenga cinco pases donde dos de ellos refieren al mismo visitante y al mismo día del año se imprime:

```
Cannot add 1 passes
4 passes added
```

Cliente de Reserva de atracciones

La información de cuál es la acción a realizar se recibe a través de argumentos de línea de comando al llamar al *script* del cliente de reserva de atracciones y el resultado se debe imprimir en pantalla.

Por ejemplo:

```
$> ./book-cli -DserverAddress=xx.xx.xx.xx:yyyy -Daction=actionName
 [ -Dday=dayOfYear -ride=rideName -Dvisitor=visitorId -Dslot=bookingSlot
-DslotTo=bookingSlotTo ]
```

donde

- **book-cli** es el *script* que invoca al cliente de reserva de atracciones
- xx.xx.xx.xx:yyyy es la dirección IP y el puerto donde está publicado el servicio de reserva de atracciones
- actionName es el nombre de la acción a realizar
    - **attractions:** Deberá imprimir en pantalla el detalle de las atracciones
    - **availability:** Deberá imprimir en pantalla la disponibilidad de las atracciones para el día del año dayOfYear a partir de uno de los siguientes criterios:
        - Un slot de una atracción a partir del nombre de la atracción rideName y el slot bookingSlot
        - Un rango de slots de una atracción a partir del nombre de la atracción rideName y el rango de slots [bookingSlot, bookingSlotTo]
        - Un rango de slots de todas las atracciones del parque a partir del rango de slots [bookingSlot, bookingSlotTo].

      El orden de impresión es cronológico y desempata alfabéticamente por el nombre de la atracción. Ver detalle más abajo.
    - **book:** Deberá realizar una reserva para el visitante visitorId para visitar la atracción rideName en el día del año dayOfYear en el slot bookingSlot
    - **confirm:** Deberá confirmar una reserva para el visitante visitorId para visitar la atracción rideName en el día del año dayOfYear en el slot bookingSlot
    - **cancel:** Deberá cancelar una reserva para el visitante visitorId para visitar la atracción rideName en el día del año dayOfYear en el slot bookingSlot

Para todas las acciones se deberá imprimir en pantalla el resultado de la operación o el error correspondiente.

De esta forma,

```
$> ./book-cli -DserverAddress=10.6.0.1:50051 -Daction=book -Dday=100
-Dvisitor=ca286ef0-162a-42fd-b9ea-60166ff0a593 -Dattraction=SpaceMountain
-Dslot=15:30
```

realiza una reserva para el visitante `ca286ef0-162a-42fd-b9ea-60166ff0a593` para visitar la atracción `SpaceMountain` en el slot de las `15:45` del día `100` del año. Algunas posibles salidas son:

```
The reservation for SpaceMountain at 15:30 on the day 100 is PENDING
```

o

The reservation for SpaceMountain at 15:30 on the day 100 is CONFIRMED

Acción `availability`

Ejemplo de una salida para el slot de las 15:30 de la atracción `SpaceMountain`:

```
$> ./book-cli -DserverAddress=10.6.0.1:50051 -Daction=availability
-Dday=100 -Dattraction=SpaceMountain -Dslot=15:30
Slot  | Capacity  | Pending   | Confirmed | Attraction
15:30 |       30  |        0  |        30 | SpaceMountain
```

Ejemplo de una salida para un rango de slots `[15:30 a 16:00]` de la atracción `SpaceMountain`:

```
$> ./book-cli -DserverAddress=10.6.0.1:50051 -Daction=availability
-Dday=100 -Dattraction=SpaceMountain -Dslot=15:30 -DslotTo=16:00
Slot  | Capacity  | Pending   | Confirmed | Attraction
15:30 |       30  |        0  |        30 | SpaceMountain
15:45 |       30  |        3  |        27 | SpaceMountain
16:00 |       30  |        0  |         5 | SpaceMountain
```

Ejemplo de una salida para un rango de slots `[15:30 a 16:00]` de todas las atracciones:

```
$> ./book-cli -DserverAddress=10.6.0.1:50051 -Daction=availability
-Dday=100 -Dslot=15:30 -DslotTo=16:00
Slot  | Capacity  | Pending   | Confirmed | Attraction
15:30 |       30  |        0  |        30 | SpaceMountain
15:30 |        X  |       25  |         0 | TronLightcycle
15:45 |       30  |        3  |        27 | SpaceMountain
16:00 |       30  |        0  |         5 | SpaceMountain
16:00 |        X  |       27  |         0 | TronLightcycle
```

donde la `X` indica que la capacidad aún no fue cargada y en el parque hay dos atracciones donde ambas tienen el mismo horario de apertura y la atracción `TronLightcycle` tiene slots de 30 minutos mientras que `SpaceMountain` tiene slots de 15 minutos.

Cliente de Notificaciones de una atracción

El nombre de la atracción y el id del visitante se recibe a través de argumentos de línea de comando al llamar al *script* del cliente de notificaciones de una atracción y el resultado se debe imprimir en pantalla.

Por ejemplo:

```
$> ./notif-cli -DserverAddress=xx.xx.xx.xx:yyyy -Daction=actionName
-Dday=dayOfYear -Dride=rideName -Dvisitor=visitorId
```

donde

- **notif-cli** es el *script* que invoca a la clase del cliente de notificaciones de una atracción
- xx.xx.xx.xx:yyyy es la dirección IP y el puerto donde está publicado el servicio de notificaciones de una atracción
- actionName es el nombre de la acción a realizar
    - **follow:** Registrar a un visitante para que sea notificado de los eventos de una atracción
        - dayOfYear: el día del año (entero)
        - rideName: el nombre de la atracción sobre la cual ser notificado (String)
        - visitorId: el id del visitante (UUID)
    - **unfollow:** Anular el registro, esto es dejar de recibir notificaciones, a partir de los mismos parámetros que la acción `follow`.

Se deberá imprimir en pantalla el resultado de la operación o el error correspondiente.
De esta forma,

```
$>  ./notif-cli -DserverAddress=10.6.0.1:50051 -Daction=follow -Dday=100 -Dride=SpaceMountain -Dvisitor=ca286ef0-162a-42fd-b9ea-60166ff0a593
```

registra al visitante de id `ca286ef0-162a-42fd-b9ea-60166ff0a593` para recibir notificaciones de la atracción `SpaceMountain` para el día `100` del año, utilizando el servicio remoto publicado en `10.6.0.1:50051` y cada vez que se produce un evento relacionado a la atracción en cuestión se imprime un mensaje.

Ejemplos de los eventos posibles:

The reservation for Space Mountain at 15:30 on the day 100 is PENDING.

Space Mountain announced slot capacity for the day 100: 15 places.

The reservation for Space Mountain at 15:30 on the day 100 is CONFIRMED.

The reservation for Space Mountain at 15:30 on the day 100 is CANCELLED.

The reservation for Space Mountain at 15:30 on the day 100 was moved to 15:45 and is PENDING.

El cliente de notificaciones de una atracción deberá finalizar su ejecución:

- En caso de que se invoque la acción **unfollow**
- Luego de que la reserva se haya confirmado o cancelado

Cliente de Consulta

Deberá dejar en archivos TXT los resultados de las consultas realizadas. Las consultas posibles son las siguientes:

Consulta 1: Capacidad sugerida

Cada atracción debe contener los siguientes campos:

- El nombre de la atracción
- La capacidad sugerida
- El slot correspondiente a esa cantidad máxima (en formato HH:MM). En caso de haber dos o más, indicar el menor en orden cronológico

Para cada atracción se debe respetar el orden descendente por la capacidad sugerida y desempata alfabéticamente por nombre de atracción.

❏ Salida de ejemplo:

```
Slot  | Capacity | Attraction
13:00 |       27 | Tron Lightcycle
12:45 |       23 | Space Mountain
09:00 |        0 | The Hall of Presidents
```

Consulta 2: Reservas confirmadas

Cada reserva debe contener los siguientes campos:

- El slot de la reserva (en formato HH:MM)
- El id del visitante de la reserva (en formato UUID)
- El nombre de la atracción

Para cada reserva se debe respetar el orden alfabético por nombre de atracción y desempata cronológicamente por slot y luego alfabéticamente por id del visitante.

❏ Salida de ejemplo:

```
Slot  | Visitor                              | Attraction
15:30 | ca286ef0-162a-42fd-b9ea-60166ff0a593 | Space Mountain
13:00 | ca286ef0-162a-42fd-b9ea-60166ff0a593 | Tron Lightcycle
13:00 | 2af16ea7-4af1-47f6-bf46-8515de5a500f | Tron Lightcycle
```

La información de cuál es la consulta a realizar se recibe a través de argumentos de línea de comando al llamar al *script* del cliente de consulta y el resultado se debe escribir en un archivo.

Por ejemplo:

```
$> ./query-cli -DserverAddress=xx.xx.xx.xx:yyyy -Daction=actionName
-Dday=dayOfYear -DoutPath=output.txt
```

donde

- **`query-cli`** es el *script* que invoca a la clase del cliente de consulta
- `xx.xx.xx.xx:yyyy` es la dirección IP y el puerto donde está publicado el servicio de consulta
- `dayOfYear` es el día del año (entero)
- `actionName` es el nombre de la acción a realizar
  - **`capacity`**: Resuelve la **Consulta 1**
  - **`confirmed`**: Resuelve la **Consulta 2**
- `output.txt` es el *path* del archivo de salida con los resultados de la consulta elegida

De esta forma,

```
$> ./query-cli -DserverAddress=10.6.0.1:50051 -Daction=capacity -Dday=100
-DoutPath=query1.csv
```

realiza la **Consulta 1** para el día del año `100` utilizando el servicio remoto publicado en `10.6.0.1:50051` y deja en `query1.csv` los resultados obtenidos según el formato arriba mencionado.

En caso de que ocurra un error o que no exista ninguna atracción o reserva confirmada el cliente de consulta deberá imprimir en pantalla un mensaje y no generar el archivo de salida.

Hechos y Consideraciones

Para simplificar el desarrollo se pueden tomar los siguientes considerandos como ciertos:

- Se asume que el formato y contenido de los archivos de entrada es correcto y no es necesario validarlo
- No es necesario que tenga persistencia. Al reiniciar el sistema se comienza de cero la operación del mismo
- La cantidad de días del año es siempre 365

Requisitos

Se requiere implementar:

- Todos los servicios deben ser implementados utilizando gRPC, teniendo en cuenta que los servicios deben poder atender pedidos de clientes de manera concurrente y responder a lo indicado en este enunciado
- Los clientes indicados cada uno como una aplicación diferente

Muy Importante:

➔ **Respetar exactamente los nombres de los *scripts*, los nombres de los archivos de salida y el orden y formato de los parámetros del *scripts***

➔ En todos los `pom.xml` que entreguen deberán definir el `artifactId` de acuerdo a la siguiente convención: “`tpe1-gX-Z`” donde X es el número de grupo y Z es `parent`, `api`, `server` o `client`. Por ejemplo: **`<artifactId>`**`tpe1-g7-api`**`</artifactId>`**

➔ En todos los `pom.xml` que entreguen deberán incluir el tag `name` con la siguiente convención: “`tpe1-gX-Z`” donde X es el número de grupo y Z es `parent`, `api`, `server` o `client`. Por ejemplo: **`<name>`**`tpe1-g7-api`**`</name>`**

➔ La implementación debe **respetar exactamente el formato de salida enunciado**

Material a entregar

Cada grupo deberá subir al **Campus ITBA** un archivo compactado conteniendo:

- El **código fuente** de la aplicación:
  Utilizando el arquetipo de Maven utilizado en las clases
  - Con una correcta separación de las clases en los módulos *api*, *client* y *server*
  - Un README indicando cómo preparar el entorno a partir del código fuente para correr el servidor y los cuatro clientes
  - El directorio oculto `.git/` donde se encuentra la historia de commits y modificaciones
  - **No se deben entregar los binarios**
- Un **documento breve** (no más de cuatro carillas) explicando:
  - Decisiones de diseño e implementación de los servicios
  - Criterios aplicados para el trabajo concurrente
  - Potenciales puntos de mejora y/o expansión

Corrección

El trabajo no se considerará aprobado si:

- No se entregó el trabajo práctico en tiempo y forma
- No se entrega alguno de los materiales solicitados en la sección anterior
- El código no compila utilizando Maven en consola (de acuerdo a lo especificado en el README a entregar)
- El servicio no inicia cuando se siguen los pasos del README
- Los clientes no corren al seguir los pasos del README

Si nada de esto se cumple, se procederá a la corrección donde se tomará en cuenta:

- Que los servicios y clientes funcionen correctamente según las especificaciones dadas
- El resultado de las pruebas y lo discutido en el coloquio
- La aplicación de los temas vistos en clase: Concurrencia y gRPC
- La modularización, diseño, testeo y reutilización de código
- El contenido y desarrollo del informe

Uso de Git

Es obligatorio el uso de un repositorio Git para la resolución de este TPE. Deberán crear un repositorio donde todos los integrantes del grupo colaboren con la implementación. No se aceptarán entregas que utilicen un repositorio git con un único *commit* que consista en la totalidad del código a entregar.

Los distintos *commits* deben permitir ver la evolución del trabajo, tanto grupal como individual.

Muy importante: **los repositorios creados deben ser privados, solo visibles para los integrantes del grupo y la cátedra en caso de que se lo solicite específicamente.**

Cronograma

- **Presentación del Enunciado: <u>miércoles 30/08</u>**
- **Entrega del trabajo: Estará disponible hasta el <u>jueves 14/09 a las 23:59</u>** la actividad "Entrega TPE 1" localizada en la sección Contenido / Evaluación / TPE 1. En la misma deberán cargar el paquete con el **código fuente** y el **documento**
- **El día <u>miércoles 04/10 a las 18:00</u>** cada grupo tendrá un espacio de 15 minutos para un coloquio. Durante el mismo se les hará una devolución del trabajo, indicando la nota y los principales errores cometidos. A criterio de la cátedra también se podrán realizar preguntas sobre la implementación. Opcionalmente se les podrá solicitar la ejecución de la aplicación a la cátedra para revisar el funcionamiento de alguna funcionalidad. Para ello es necesario que un integrante del equipo tenga el sistema remoto “levantado”.
- **El día del recuperatorio será el <u>miércoles 22/11</u>**
- **<u>No se aceptarán entregas pasado el día y horario establecido como límite</u>**

Dudas sobre el TPE

Las mismas deben volcarse en los **Debates** del Campus ITBA.

Recomendaciones

- **Clases y métodos útiles para consultar**
  - `java.nio.file.Files.lines`
  - `java.nio.file.Files.write`
  - `java.time.LocalTime`